Fundamentos de Algoritmia

Grado en Informática. Grupos A, C, DG, F

Convocatoria extraordinaria, 7 de Septiembre de 2020.

Normas de realización del examen

1. Debes programar soluciones para cada uno de los tres ejercicios, probarlas y entregarlas en el juez automático accesible en la dirección http://exacrc/domjudge/team.
2. Escribe comentarios que expliquen tu solución, justifiquen por qué se ha hecho así y ayuden a entenderla. Calcula el coste en tiempo de todas las funciones que implementes.
3. En el juez te identificarás con el nombre de usuario y contraseña que has recibido al comienzo del examen. El nombre de usuario y contraseña que has estado utilizando durante la evaluación continua **no** son válidos.
4. Escribe tu **nombre y apellidos** en un comentario en la primera línea de cada fichero que subas al juez.
5. Tus soluciones serán evaluadas por la profesora independientemente del veredicto del juez automático. Para ello, la profesora tendrá en cuenta **exclusivamente** el último envío que hayas realizado de cada ejercicio.

1. **(3 puntos)** Un patrocinador está buscando jugadores de dardos. Observa a los jugadores y anota los puntos que obtienen en una secuencia. Ofrece un patrocinio a un jugador si obtiene una cantidad de puntos estrictamente superior a 1.5 multiplicado por la media de puntos obtenidos por los jugadores anteriores.

    1. (0,75 puntos) *Especifica formalmente* una función que dado un vector de enteros de longitud ≥ 0, que corresponde a la secuencia de puntos obtenida por los jugadores, devuelva el número de jugadores patrocinados.
    2. (1,5 puntos) *Diseña e implementa* un algoritmo iterativo que resuelva el problema propuesto.
    3. (0,5 puntos) Escribe el *invariante* del bucle que permite demostrar la corrección del mismo y proporciona una función de cota.
    4. (0,25 puntos) Indica el *coste asintótico* del algoritmo en el caso peor y justifica adecuadamente tu respuesta.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba. Cada caso de prueba contendrá el valor del número de jugadores n y a continuación los puntos obtenidos por los jugadores.

Salida

Por cada caso de prueba el programa escribirá una línea con el número de jugadores patrocinados.

Entrada de ejemplo

```
4
1 10
3 1 5 10
3 5 6 7
4 15 5 15 18
```

Salida de ejemplo

```
1
3
1
2
```

2.(2,5 puntos) Una compañía ha decidido premiar a algunos de sus empleados con una condiciones privilegiadas. Dichos empleados incrementarán sus salarios todos los años y el incremento de cada año será estrictamente mayor que el incremento del año anterior.

1. (1,75 puntos) Diseña e implementa una función recursiva eficiente que reciba un vector no vacío de salarios y un incremento de salario ≥ 0 que el empleado ha obtenido con seguridad algún año, y devuelva el año en que el empleado ha obtenido dicho incremento de salario con respecto al anterior. Los años se cuentan desde el año en que comenzó el contrato del empleado, que se considera el año 0, en el cual se considera que el incremento es 0 porque no ha trabajado antes en esa empresa.
2. (0,75 puntos) Escribe la recurrencia que corresponde al coste de la función indicando claramente cuál es el tamaño del problema. Indica también a qué orden de complejidad asintótica pertenece dicho coste.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba. Cada caso de prueba consta de dos líneas: la primera contiene el incremento de salario que se busca y el número de años que el empleado lleva trabajando. La segunda línea contiene los salarios de cada año que ha trabajado el empleado.

Salida

Por cada caso de prueba el programa escribirá el año en que el empleado obtuvo el incremento de salario buscado.

Entrada de ejemplo

```
3
8 10
1000 1001 1003 1008 1015 1023 1040 1080 1130 1190
2 10
1000 1001 1003 1008 1015 1023 1040 1080 1130 1190
60 10
1000 1001 1003 1008 1015 1023 1040 1080 1130 1190
```

Salida de ejemplo

```
5
2
9
```

3. (4,5 puntos) La Universidad de FALbridge va a reabrir sus puertas a profesores, personal de administración y servicios, y estudiantes. Para poder cumplir con las medidas sanitarias contra el virus, va a repartir mascarillas a la entrada de sus $n \geq 1$ facultades. Se ha establecido que cada facultad va a necesitar una cantidad determinada > 0 de mascarillas. La Universidad se ha puesto en contacto con $m \geq n$ suministradores. Cada suministrador dispone de un determinado stock de mascarillas y las ofrece a un determinado precio (todas las de un suministrador tienen el mismo precio). Cada suministrador puede suministrar mascarillas como máximo a una facultad, pero una facultad puede cubrir sus necesidades con uno o varios suministradores. El objetivo es minimizar el coste total de las mascarillas en toda la Universidad, de forma que queden cubiertas las necesidades de todas las facultades. Por supuesto, la Universidad solo pagará las mascarillas que necesite.

- (0,5 puntos) Define el *espacio de soluciones* e indica cómo es el *árbol de exploración*.
- (2,5 puntos) Implementa un algoritmo de *vuelta atrás* que resuelva el problema. Explica claramente los *marcadores* que has utilizado.
- (1,5 puntos) Plantea dos posibles funciones de poda de optimalidad, razona sobre cual de ellas es mejor e impleméntala en tu algoritmo.

Entrada

La entrada comienza con una línea que contiene el número de casos de prueba. La primera línea de cada caso de prueba contendrá el valor del número de universidades n y de suministradores m. A continuación habrá una línea con las n cantidades de mascarillas que necesitan las facultades, otra con el stock de mascarillas de cada uno de los m suministradores y por último otra con los m precios por mascarilla en cada uno de los suministradores. Los suministradores estarán ordenados de menor a mayor precio.

Salida

Por cada caso de prueba el programa escribirá NO si no se puede satisfacer las necesidades de todas las facultades y una línea con el coste mínimo de comprar las mascarillas que necesitan todas las facultades en caso de que sí puedan satisfacerse.

Entrada de ejemplo

```
3
2 4
7 14
4 8 10 4
2 6 9 10
2 4
7 14
8 8 10 4
2 6 9 10
2 4
7 14
6 6 6 6
2 6 9 10
```

Salida de ejemplo

```
139
112
NO
```

NOTA: En el ejemplo, la solución al primer caso de prueba consiste en que la Facultad 0 recibe las mascarillas del suministrador 2 y la Facultad 1 las recibe de los suministradores 0, 1 y 3:

$$139 = 9 * 7 + (2 * 4 + 6 * 8 + 10 * 2)$$